WEST Search History

| Hide Items | Restore | Clear | Cancel |

DATE: Thursday, March 01, 2007

Hide?	Set Name	Query	Hit Count
		DB=PGPB,USPT,USOC; PLUR=NO; OP=OR	
☐	L18	(l13 or l14) and l17	7
☐	L17	707/200.ccls.	2852
☐	L16	l14 and ((send$ or distribut$ or transfer$ or mov$) near (file or files or folder or folders))	2
☐	L15	l13 and ((send$ or distribut$ or transfer$ or mov$) near (file or files or folder or folders))	43
☐	L14	l1 and l9	5
☐	L13	l1 and L12	84
☐	L12	((look-up$ or (look adj1 up$)) with (file or files or folder or folders))	2388
☐	L11	((lock-free or (lock adj1 free)) near (file or files or folder or folders))	3
☐	L10	((lock-free or (lock adj1 free)) with (file or files or folder or folders))	7
☐	L9	(lock-free or (lock adj1 free))	1336
☐	L8	((lock-free or (lock adj1 free)) with (look-up$ or (look adj1 up$)))	1
☐	L7	((lock-free or (lock adj1 free)) near (look-up$ or (look adj1 up$)))	1
☐	L6	L1 and ((look-up$ or (look adj1 up$)) near (file or files or folder or folders))	23
☐	L5	L1 and ((look-up$ or (look adj1 up$)) with (file or files or folder or folders))	84
☐	L4	L3 and (record or records)	307
☐	L3	L1 and (look-up$ or (look adj1 up$))	373
☐	L2	L1 and ((look-up$ or (look adj1 up$)) with (record or records))	14
☐	L1	(renam$ near (file or files or folder or folders))	1810

END OF SEARCH HISTORY

10|813,470

Subscribe (Full Service)   Register (Limited Service, Free)   Login

Search: ◉ The ACM Digital Library   ○ The Guide

rename file and look up and lock free and file systems | **SEARCH**

🖰 Feedback   Report a problem   Satisfaction survey

Terms used
rename file and **look up** and **lock free** and **file systems**

Found **98,956** of 198,146

Sort results by: [relevance ▽]

Display results: [expanded form ▽]

● Save results to a Binder
🔲 Search Tips
☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200   Result page: **1** 2 3 4 5 6 7 8 9 10   next
Best 200 shown   Relevance scale ☐ ▭ ▬ ■ ■

1 A fast file system for UNIX
Marshall K. McKusick, William N. Joy, Samuel J. Leffler, Robert S. Fabry
August 1984 **ACM Transactions on Computer Systems (TOCS)**, Volume 2 Issue 3
Publisher: ACM Press

Full text available: 🔲 pdf(1.31 MB)

Additional Information: full citation, references, citings, index terms, review

Keywords: UNIX, application program interface, file system design, file system organization, file system performance

2 A coherent distributed file cache with directory write-behind
Timothy Mann, Andrew Birrell, Andy Hisgen, Charles Jerian, Garret Swart
May 1994 **ACM Transactions on Computer Systems (TOCS)**, Volume 12 Issue 2
Publisher: ACM Press

Full text available: 🔲 pdf(3.21 MB)

Additional Information: full citation, abstract, references, citings, index terms, review

Extensive caching is a key feature of the Echo distributed file system. Echo client machines maintain coherent caches of file and directory data and properties, with write-behind (delayed write-back) of all cached information. Echo specifies ordering constraints on this write-behind, enabling applications to store and maintain consistent data structures in the file system even when crashes or network faults prevent some writes from being completed. In this paper we describe ...

Keywords: coherence, file caching, write-behind

3 Distributed operating systems
Andrew S. Tanenbaum, Robbert Van Renesse
December 1985 **ACM Computing Surveys (CSUR)**, Volume 17 Issue 4
Publisher: ACM Press

Full text available: 🔲 pdf(5.49 MB)

Additional Information: full citation, abstract, references, citings, index terms, review

Distributed operating systems have many aspects in common with centralized ones, but

10\813,470

they also differ in certain ways. This paper is intended as an introduction to distributed operating systems, and especially to current university research about them. After a discussion of what constitutes a distributed operating system and how it is distinguished from a computer network, various key design issues are discussed. Then several examples of current research projects are examined in some detail ...

4   Deciding when to forget in the Elephant file system

Douglas S. Santry, Michael J. Feeley, Norman C. Hutchinson, Alistair C. Veitch, Ross W. Carton, Jacob Ofir

December 1999 **ACM SIGOPS Operating Systems Review , Proceedings of the seventeenth ACM symposium on Operating systems principles SOSP '99**, Volume 33 Issue 5

Publisher: ACM Press

Full text available: pdf(1.61 MB)   Additional Information: full citation, abstract, references, citings, index terms

Modern file systems associate the deletion of a file with the immediate release of storage, and file writes with the irrevocable change of file contents. We argue that this behavior is a relic of the past, when disk storage was a scarce resource. Today, large cheap disks make it possible for the file system to protect valuable data from accidental delete or overwrite.This paper describes the design, implementation, and performance of the Elephant file system, which automatically retains all impo ...

5   Charles W. Bachman interview: September 25-26, 2004; Tucson, Arizona

Thomas Haigh

January 2006 **ACM Oral History interviews**

Publisher: ACM Press

Full text available: pdf(761.66 KB)   Additional Information: full citation, abstract

Charles W. Bachman reviews his career. Born during 1924 in Kansas, Bachman attended high school in East Lansing, Michigan before joining the Army Anti Aircraft Artillery Corp, with which he spent two years in the Southwest Pacific Theater, during World War II. After his discharge from the military, Bachman earned a B.Sc. in Mechanical Engineering in 1948, followed immediately by an M.Sc. in the same discipline, from the University of Pennsylvania. On graduation, he went to work for Do ...

6   The Cedar file system

David K. Gifford, Roger M. Needham, Michael D. Schroeder

March 1988 **Communications of the ACM**, Volume 31 Issue 3

Publisher: ACM Press

Full text available: pdf(1.26 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

The Cedar File System (CFS) is a workstation file system that provides access to both a workstation's local disk and to remote file servers via a single hierarchical name space. CFS supports a group of cooperating programmers by allowing them to both manage local naming environments and to share consistent versions of collections of software.

7   Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research CASCON '97**

Publisher: IBM Press

Full text available: pdf(4.21 MB)   Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the

execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

8   A caching file system for a programmer's workstation

Michael D. Schroeder, David K. Gifford, Roger M. Needham

December 1985 **ACM SIGOPS Operating Systems Review , Proceedings of the tenth ACM symposium on Operating systems principles SOSP '85**, Volume 19 Issue 5

Publisher: ACM Press

Full text available: ⬛ pdf(768.75 KB)   Additional Information: <u>full citation</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

9   Register Packing: Exploiting Narrow-Width Operands for Reducing Register File Pressure

Oguz Ergin, Deniz Balkan, Kanad Ghose, Dmitry Ponomarev

December 2004 **Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture MICRO 37**

Publisher: IEEE Computer Society

Full text available: ⬛ pdf(224.06 KB)   Additional Information: <u>full citation</u>, <u>abstract</u>

A large percentage of computed results have fewer significant bits compared to the full width of a register. We exploit this fact to pack multiple results into a single physical register to reduce the pressure on the register file in a superscalar processor. Two schemes for dynamically packing multiple "narrow-width" results into partitions within a single register are evaluated. The first scheme is conservative and allocates a full-width register for a computed result. If the computed result tu ...

10 The *Conquest* file system: Better performance through a disk/persistent-RAM hybrid design

An-I Andy Wang, Geoff Kuenning, Peter Reiher, Gerald Popek

August 2006 **ACM Transactions on Storage (TOS)**, Volume 2 Issue 3

Publisher: ACM Press

Full text available: ⬛ pdf(1.34 MB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>index terms</u>

Modern file systems assume the use of disk, a system-wide performance bottleneck for over a decade. Current disk caching and RAM file systems either impose high overhead to access memory content or fail to provide mechanisms to achieve data persistence across reboots.The *Conquest* file system is based on the observation that memory is becoming inexpensive, which enables all file system services to be delivered from memory, except for providing large storage capacity. Unlike caching, *Con* ...

Keywords: *Persistent RAM, file systems, performance measurement, storage management*

11 A high performance multi-structured file system design

Keith Muller, Joseph Pasquale

September 1991 **ACM SIGOPS Operating Systems Review , Proceedings of the thirteenth ACM symposium on Operating systems principles SOSP '91**, Volume 25 Issue 5

Publisher: ACM Press

Full text available: ⬛ pdf(1.40 MB)   Additional Information: <u>full citation</u>, <u>abstract</u>, <u>references</u>, <u>citings</u>, <u>index terms</u>

File system I/O is increasingly becoming a performance bottleneck in large distributed computer systems. This is due to the increased file I/O demands of new applications, the inability of any single storage structure to respond to these demands, and the slow decline of, disk access times (latency and seek) relative to the rapid increase in CPU speeds, memory size, and network bandwidth. We present a *multi-structured file system* designed for high bandwidth I/O and fast response. Our design ...

12 4.2BSD and 4.3BSD as examples of the UNIX system

John S. Quarterman, Abraham Silberschatz, James L. Peterson
December 1985 **ACM Computing Surveys (CSUR)**, Volume 17 Issue 4
Publisher: ACM Press.

Full text available: pdf(4.07 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

This paper presents an in-depth examination of the 4.2 Berkeley Software Distribution, Virtual VAX-11 Version (4.2BSD), which is a version of the UNIX Time-Sharing System. There are notes throughout on 4.3BSD, the forthcoming system from the University of California at Berkeley. We trace the historical development of the UNIX system from its conception in 1969 until today, and describe the design principles that have guided this development. We then present the internal data structures and ...

13 Versatility and Unix semantics in namespace unification

Charles P. Wright, Jay Dave, Puja Gupta, Harikesavan Krishnan, David P. Quigley, Erez Zadok, Mohammad Nayyer Zubair
February 2006 **ACM Transactions on Storage (TOS)**, Volume 2 Issue 1
Publisher: ACM Press

Full text available: pdf(317.82 KB)   Additional Information: full citation, abstract, references, citings, index terms

Administrators often prefer to keep related sets of files in different locations or media, as it is easier to maintain them separately. Users, however, prefer to see all files in one location for convenience. One solution that accommodates both needs is virtual namespace unification---providing a merged view of several directories without physically merging them. For example, namespace unification can merge the contents of several CD-ROM images without unpacking them, merge binary directories fr ...

Keywords: Namespace management, directory merging, snapshotting, stackable file systems, unification

14 A low-bandwidth network file system

Athicha Muthitacharoen, Benjie Chen, David Mazières
October 2001 **ACM SIGOPS Operating Systems Review , Proceedings of the eighteenth ACM symposium on Operating systems principles SOSP '01**, Volume 35 Issue 5
Publisher: ACM Press

Full text available: pdf(1.29 MB)   Additional Information: full citation, abstract, references, citings, index terms

Users rarely consider running network file systems over slow or wide-area networks, as the performance would be unacceptable and the bandwidth consumption too high. Nonetheless, efficient remote file access would often be desirable over such networks---particularly when high latency makes remote login sessions unresponsive. Rather than run interactive programs such as editors remotely, users could run the programs locally and manipulate remote files through the file system. To do so, however, wo ...

15 Interactive Editing Systems: Part II

Norman Meyrowitz, Andries van Dam
September 1982 **ACM Computing Surveys (CSUR)**, Volume 14 Issue 3
Publisher: ACM Press
Full text available: pdf(9.17 MB)   Additional Information: full citation, references, citings, index terms

16 Evaluation of two relational database management systems: UNIFY and iDB

Lindsay McDermid
May 1986 **ACM SIGSMALL/PC Notes**, Volume 12 Issue 2
Publisher: ACM Press
Full text available: pdf(3.41 MB)   Additional Information: full citation, abstract, index terms

> The following document is an evaluation and comparison of two relational database management systems: UNIFY and iDB. UNIFY Release 3.1 runs on the NCR Tower iDB runs a version of Mistress under iDIS Release 1.6 on the Intel 310.

17 Fortran 8X draft

Loren P. Meissner
December 1989 **ACM SIGPLAN Fortran Forum**, Volume 8 Issue 4
Publisher: ACM Press
Full text available: pdf(21.36 MB)   Additional Information: full citation, abstract, index terms

> **Standard Programming Language Fortran.** This standard specifies the form and establishes the interpretation of programs expressed in the Fortran language. It consists of the specification of the language Fortran. No subsets are specified in this standard. The previous standard, commonly known as "FORTRAN 77", is entirely contained within this standard, known as "Fortran 8x". Therefore, any standard-conforming FORTRAN 77 program is standard conforming under this standard. New features can b ...

18 Using model checking to find serious file system errors

Junfeng Yang, Paul Twohey, Dawson Engler, Madanlal Musuvathi
November 2006 **ACM Transactions on Computer Systems (TOCS)**, Volume 24 Issue 4
Publisher: ACM Press
Full text available: pdf(534.00 KB)   Additional Information: full citation, abstract, references, index terms

> This article shows how to use model checking to find serious errors in file systems. Model checking is a formal verification technique tuned for finding corner-case errors by comprehensively exploring the state spaces defined by a system. File systems have two dynamics that make them attractive for such an approach. First, their errors are some of the most serious, since they can destroy persistent data and lead to unrecoverable corruption. Second, traditional testing needs an impractical, expon ...
>
> **Keywords:** Model checking, crash, file system, journaling, recovery

19 A framework for the assessment of operating systems for small computers

Hossein Saiedian, Munib Siddiqi
April 1996 **ACM SIGICE Bulletin**, Volume 21 Issue 4
Publisher: ACM Press
Full text available: pdf(1.89 MB) · Additional Information: full citation, abstract, references, index terms

> A number of high performance operating systems are now available for small computers on different hardware platforms. These operating systems offer many advanced features formerly reserved for their workstation and minicomputer counterparts. This article surveys the most widely used of such operating systems, namely OS/2, Windows NT,

Linux and Macintosh System 7.5. It provides an account on the history, design objectives and evolution of these operating systems and discusses their key features, ...

Keywords: CP/M, DOS, Linux, Macintosh, Microcomputers, OS/2, Operating Systems, Small Computer Systems, Windows, Windows NT

20 Decentralized storage systems: Taming aggressive replication in the Pangaea wide-area file system

Yasushi Saito, Christos Karamanolis, Magnus Karlsson, Mallik Mahalingam

December 2002 **ACM SIGOPS Operating Systems Review**, Volume 36 Issue SI

Publisher: ACM Press

Full text available: pdf(1.93 MB)   Additional Information: full citation, abstract, references

Pangaea is a wide-area file system that supports data sharing among a community of widely distributed users. It is built on a symmetrically decentralized infrastructure that consists of commodity computers provided by the end users. Computers act autonomously to serve data to their local users. When possible, they exchange data with nearby peers to improve the system's overall performance, availability, and network economy. This approach is realized by aggressively creating a replica of a file w ...

Results 1 - 20 of 200   Result page: **1** 2 3 4 5 6 7 8 9 10   next